Handbuch

XMLRPC- / JSONRPC- / SOAP-API

1 Allgemeines

content.de bietet seinen Kunden eine API an, über die Sie von externen Systemen aus auf das content.de System zugreifen können. Der Zugriff erfolgt wahlweise über eine XMLRPC, JSONRPC oder SOAP Schnittstelle. Die Schnittstellen sind zu erreichen über:

- https://www.content.de/api/clientservices.php?wsdl
- https://www.content.de/api/xmlrpc.php
- https://www.content.de/api/json-rpc.php

Im Folgenden beziehen sich die Beispiele auf den Aufruf der SOAP-API über PHP. Die XMLRPC-Schnittstellenaufrufe erfolgen analog zu den SOAP-Beispielen. Exemplarisch wird zum ersten Beispiel auch die XMLRPC-Variante mit angegeben.

2 Authentifizierung

Um nicht bei jedem Request alle Login-Daten übertragen zu müssen, authentifiziert man sich einmalig für eine Arbeitssitzung und erhält einen Session-Hash, der bei allen folgenden Aufrufen zur Authentifizierung verwendet wird.

Beispiel SOAP:

```
define ("WSDL_SERVER", "https://www.content.de/api/clientservices.php?wsdl");
header("content-type:text/html; charset=utf-8");
$soap=new SoapClient(WSDL_SERVER);
$hash=$soap->login('username@ihredomain.de','geheim');
```

Beispiel XMLRPC über cURL:

```
define('CONTENT_DE_XMLRPC_URL', 'https://www.content.de/api/xmlrpc.php');
$sRpcRequest = xmlrpc_encode_request('content.login',
    array('username@ihredomain.de','geheim'));
$rCurl = curl_init(CONTENT_DE_XMLRPC_URL);
$aHeaders = array('content-type: text/xml');
curl_setopt_array($rCurl, array(
    CURLOPT_POSTFIELDS => $sRpcRequest,
    CURLOPT_FOLLOWLOCATION => true,
    CURLOPT_RETURNTRANSFER => true,
    CURLOPT_TIMEOUT => 50,
    CURLOPT_HTTPHEADER => $aHeaders
));
$sResponse = curl_exec($rCurl);
$aResponse = xmlrpc_decode($sResponse);
```

Über die Funktion logout() beendet man die Arbeitssession und invalidiert den Session-Hash.

3 Einheitliches Result Objekt

Um das Handling der zurückgegebenen Daten zu vereinfachen, geben alle Funktionen ein einheitliches Result Objekt zurück. Es enthält folgende Werte:

- `NumRecords: integer`
- `Records: array()`
- `HasError: bool`
- `ErrorMsg: string`

Das Feld `NumRecords` gibt die Anzahl der übermittelten Werte an. `Records` enthält ein Array der zu übermittelnden Werte.

Im Fall eines Fehlers wird die Variable `HasError` auf `TRUE` gesetzt. In diesem Fall findet sich in `ErrorMsg` eine Fehlermeldung.

Achtung: Bei einigen Methoden, die keine reine „Datenabfrage“ darstellen, wie z.B. das Anlegen eines Auftrags, wird kein Array mit Daten zurückgegeben, sondern ein einzelner Wert, wie z.B. die ID des angelegten Auftrags.

4 Exception

In einem außergewöhnlichen Fehlerfall wird eine SOAP-Exception erzeugt. Diese Exception können Sie wie folgt abfangen:

```
define ("WSDL_SERVER", "https://www.content.de/api/clientservices.php?wsdl");
header("content-type:text/html; charset=utf-8");
try{
    $soap=new SoapClient(WSDL_SERVER);
    hash=$soap->login('username@ihredomain.de','geheim');
}catch(Exception $e)
{
    var_dump($e);
}
```

5 Daten abfragen

Zur Einstellung von Aufträge kann es notwendig sein, zuvor bestimmte Daten abzufragen, die beispielsweise als Schlüsselwerte, wie die Schwierigkeitsstufen bei der Auftragsvergabe mit angegeben werden müssen.

5.1 Kontostand

Um den Kontostand abzufragen, ruft man die Funktion getBalance() auf und übergibt als einzigen Parameter den Session-Hash, der der Rückgabewert der oben gezeigten Funktion ist.

Beispiel:

```
$oResult=$soap->getBalance($hash)
echo $oResult->Records['balance'];
```

5.2 Projekte

Um einen Auftrag anzulegen, muss dieser einem Projekt zugeordnet werden. Dazu wird die Projekt-ID übergeben. Eine Liste der eigenen Projekte erhält man über die Funktion getProjects(), die als einzigen Parameter wiederum den Session-Hash benötigt.

```
$oResult=$soap->getProjects($hash);
```

5.3 Schwierigkeitsstufen

Ebenso wird für jeden Auftrag die Vorgabe einer Schwierigkeitsstufe verlangt, die der Sterne-Einstufung eines Auftrags entspricht.

```
$oResult=$soap->getLevels($hash);
```

5.4 Kategorien

Die IDs der möglichen Kategorien sind über die Funktion getCategories() zu ermitteln.

```
$oResult=$soap->getCategories($hash);
```

5.5 Textarten

Die IDs der möglichen Textarten sind über die Funktion getTextTypes() zu ermitteln. Textarten werden bei der Auftragsvergabe über die API von dem zugeordneten Projekt geerbt. Der Parameter für das setzen der Textart lautet `text_type`. Vgl dazu 6.5 . Alternativ kann die Textart als optionaler Parameter bei der Auftragserstellung direkt gesetzt werden.

```
$oResult=$soap->getTextTypes($hash);
```

5.6 Templates

Ein wichtiger Bestandteil eines Auftrags ist das Autorenbriefing. Für die Autorenbriefings lassen sich über das content.de Frontend Templates anlegen, die man über die Funktion getTemplates() abrufen kann.

```
$oResult=$soap->getTemplates($hash);
```

5.7 Autoren

Um eine Direct Order an einen bestimmten Autor zu vergeben, benötigt man die ID des Autors. Den Inhalt seines Adressbuchs kann man sich über die Funktion getAddressBook () abrufen.

```
$oResult=$soap-> getAddressBook ($hash);
```

5.8 Gruppen

Um eine Group Order zu erteilen, werden die IDs der Gruppen benötigt, die man über das Frontend mit dem Tagging-Mechanismus angelegt hat.

```
$oResult=$soap->getGroups($hash);
```

6 Aufträge einstellen

Je nach Art des Auftrags werden unterschiedliche Funktionen verwendet, um die Aufträge einzustellen. Zahlreiche notwendige Parameter sind jedoch bei allen Auftragsarten gleich:

- string sSessionHash: Hash zur Authentifizierung
- string sTitel: Auftragstitel
- array aKeywords: Array mir Keywordphrasen als Zeichenketten
- string sDesciption: Autorenbriefing
- string sProject: Projekt ID
- integer iCategory: ID der gewünschten Kategorie
- integer iDuration: Bearbeitungszeit in Tagen (Default 3)
- integer iLevel: Schlüssel der Schwierigkeitsstufe (Default 10 => 4 Sterne)

- integer iMin: minimale Wortanzahl (min. 100, Default 200)
- integer iMax: maximale Wortanzahl (Default 500)
- integer iMinDensity: minimale Keyworddichte (Default 1)
- integer iMaxDensity: maximale Keyworddichte (Default 3)
- string sExternalId: Eine ID aus Ihrem System, mit der Sie den Auftrag indentifizieren können, z.B. eine Artikelnummer

Als Rückgabewert erhält man die ID des Auftrags im content.de System. Sie wird benötigt, um einzelne Aufträge später wieder gezielt abzurufen.

6.1 Open Order erstellen

Zur Erstellung von Open Orders werden die zuvor beschriebenen Parameter wie folgt übergeben:

```
$soap->createOpenOrder($sSessionHash, $sTitle, $aKeywords, $sDesciption,
$sProject, $iCategory, $iDuration, $iLevel, $iMin, $iMax, $iMinDensity,
$iMaxDensity, $sExternalId, $iTextType);
```

Die Parameter `$iDuration, $iMin, $iMax, $iMinDensity, $iMaxDensity, $sExternalId, $iTextType` sind optional.

6.2 Group Order erstellen

Bei Group Orders ist zusätzlich der ganzzahlige Parameter iGroup erforderlich, der die Gruppen ID enthält:

```
$soap->createGroupOrder($sSessionHash, $sTitle, $aKeywords, $sDesciption,
$sProject, $iCategory, $iGroup, $iDuration, $iLevel, $iMin, $iMax,
$iMinDensity, $iMaxDensity, $sExternalId, $iTextType);
```

Die Parameter `$iDuration, $iMin, $iMax, $iMinDensity, $iMaxDensity, $sExternalId, $iTextType` sind optional.

6.3 Direct Order erstellen

Bei der Direct Order entfällt der Parameter iLevel, dafür wird die ID des Autors benötigt:

```
$soap->createDirectOrder($sSessionHash, $sTitle, $aKeywords, $sDesciption,
$sProject, $sContractor, $iCategory, $iDuration, $iMin, $iMax, $iMinDensity,
$iMaxDensity, $sExternalId, $iTextType);
```

Die Parameter `$iDuration, $iMin, $iMax, $iMinDensity, $iMaxDensity, $sExternalId, $iTextType` sind optional.

6.4 Projekte anlegen

Projekte können durch Übergabe des Projektnamens erzeugt werden:

```
$soap->createProject($sSessionHash, $sProjectName)
```

6.5 Projekteeinstellungen festlegen

Bestimmte Projekteinstellungen können ebenfalls über die API vorgenommen werden. Die einzelnen möglichen Parameter erfragen Sie bitte bei unserer Technik.

```
$soap->setProjectSettings ($sSessionHash, $sProjectId, array(key => value))
```

7 Aufträge abholen, bewerten und archivieren

7.1 Alle zur Abnahme fertigen Aufträge eines Projektes abrufen

Eine Liste aller aktuell zur Abnahme fertigen Texte ist über die Funktion getSubmittedOrdersByProject erhältlich. Neben dem Session Hash ist die Projekt-ID als Übergabeparameter notwendig.

```
$soap->getSubmittedOrdersByProject($sSessionHash, $sProject);
```

7.2 Einzelnen zur Abnahme bereiten Auftrag abrufen

Einzelne Aufträge können über ihre ID abgerufen werden:

```
$soap->getOrderById($sSessionHash, $sOrderId);
```

7.3 Auftrag annehmen und Bewerten

Nachdem der Autor einen Text zur Abnahme eingereicht hat, können Sie prüfen, ob der Text Ihren Vorgaben entspricht. Wenn Sie den Text akzeptieren wollen, rufen Sie die Funktion rateAndAcceptOrder() auf. Sie können dem Autor noch einen kurzen Bewertungstext mitgeben und weiterhin vier Bewertungsparameter für Inhalt, Form, Lesbarkeit sowie Kommunikation & Termintreue angeben. Der Defaultwert 0 bedeutet, dass das Ergebnis Ihre Anforderungen und Erwartungen genau erfüllt hat. Sie können positiv und negativ mit den Werten 1 und 2 bzw. -1 und -2 abweichen.

```
$soap->rateAndAcceptOrder($sSessionHash, $sOrderId, $sText, $content_rating,
$form_rating, $readability_rating, $communication_rating, $bExport)
```

7.4 Alle Aufträge eines Projektes abrufen

Um alle akzeptierten Texte eines Projektes abzurufen, verwenden Sie die Funktion

getRatedOrdersByProject(), die neben dem Session-Hash die Projekt ID als Parameter benötigt. Es werden nur nicht-archivierte Aufträge abgerufen

```
$soap->getRatedOrdersByProject($sSessionHash, $sProjectId);
```

7.5 Archivierte Aufträge eines Projektes abrufen

Um alle archivierten Texte eines Projektes abzurufen, verwenden Sie die Funktion

getArchivedOrdersByProject(), die neben dem Session-Hash die Projekt ID als Parameter benötigt.

```
$soap->getArchivedOrdersByProject($sSessionHash, $sProjectId);
```

7.6 Überarbeitung beauftragen

Sollte ein Auftrag nicht Ihren Wünschen entsprechend bearbeitet worden sein (wenn beispielsweise Anforderungen aus dem Autorenbriefing nicht beachtet wurden), können Sie den Auftrag zur Überarbeitung an den Autor zurückgeben. Neben dem Session-Hash und der Order-ID müssen Sie eine Begründung mit Hinweisen zur Überarbeitung angeben.

```
$soap->requestRevision($sSessionHash, $sOrderId, $sText);
```

7.7 Ablehnung beantragen

Sollte ein Text nach einer oder mehreren Überarbeitungen immer noch nicht Ihren Erwartungen entsprechen, können Sie eine Ablehnung beantragen. Geben Sie auch hier eine Begründung an. Das content.de - Team prüft den Fall anschließend umgehend.

```
$soap->requestRejection($sSessionHash, $sOrderId, $sText);
```

7.8 Auftrag archivieren

Akzeptierte Aufträge können ins Archiv verschoben werden. Mit dem Parameterwert 1 des letzten Parameters werden die die Aufträge archiviert, der Parameterwert 0 holt den Auftrag aus dem Archiv zurück.

```
$soap->archiveOrder($sSessionHash, $sOrderId, $bArchiv);
```

7.9 Auftrag stornieren

Noch offene Open- und Group Orders können storniert werden.

```
$soap->cancelOrderById($sSessionHash, $sOrderId);
```

8 Nachrichten

Über die API kann auch auf das Nachrichtensystem zugegriffen werden. Sinnvoll ist es, die Kommunikation immer in Verbindung mit dem konkreten Auftrag abzuwickeln und die Nachrichten auch auftragsbezogen abzufragen, bzw. zu versenden.

8.1 auftragsbezogene Nachrichten abrufen

Abfragen einer Nachricht mittels Order ID. Wird der optionale dritte Parameter auf TRUE gesetzt, werden auch die Nachrichten von Autoren, die den Auftrag vor dem aktuellen Autor bearbeitet haben, mit ausgegeben

$soap->getOrderMessages($sSessionHash, $sOrderId, $bAddMessagesFromCancelledTexts = false)

8.2 auftragsbezogene Nachrichten senden

Bei dem Versand von auftragsbezogenen Nachrichten wird nur die Nachricht vorgegeben, der Titel wird vom System automatisch gesetzt.

$soap->writeOrderMessage($sSessionHash, $sOrderId, $sMessage)

8.3 Zugriff auf Postfächer

Es kann können auch komplette Postfächer abgerufen werden. Dazu ist die jeweilige Ordnart anzugeben. Die Abfrage sollte unter Perfomancegesichtspunkten nicht periodisch erfolgen sondern manuell angestoßen werden.

$soap->getMessages($sSessionHash, $sFolder)

$sFolder wahlweise inbox, outbov, archive oder trash

8.4 Nachricht senden

Ebenso kann eine Nachricht verschickt werden. Ist kein Bezug zu einem Auftrag möglich, wird $sOrder = "" gesetzt.

$soap->sendMessage($sSessionHash, $sTitle, $sText, $sUser, $sOrder)

8.5 Nachrichtenstatus setzen

Wurde eine Nachricht gelesen, sollte der Status mit

$soap->setMessageRead($sSessionHash, $iMessage)

Auf „gelesen“ gesetzt werden.

9 Übersetzungen

Über das content.de System lassen sich auch Übersetzungen beauftragen. Übersetzungen werden bei content.de in Paketen organisiert. Mehrere Texte können zu einem Paket zusammengefasst werden und gemeinsam beauftragt werden. Ein Paket wird dann von einem einheitlichen Übersetzungsteam übersetzt und abgerechnet. Basis der Abrechnung ist die Gesamtwortanzahl aller Texte eines Paketes. Pro Paket ist nur eine Sprachkombination möglich.

9.1 Sprachschlüssel und Preise abfragen

Die für das Anlegen eines Übersetzungspaketes notwendigen Sprachschlüssel können abgefragt

werden mit:

```
$soap->getTranslationLanguages($sSessionHash);
```

9.2 Paket anlegen

Beim Anlegen eines Pakets muss neben der Zielsprach auch die Ausgangssprache explizit festgelegt werden. Weiterhin ist neben dem Titel anzugeben, ob die Texte im 4- oder 2-Augenprinzip übersetzt werden sollen (`$iLevel=1`, für das 4 Augen Prinzip, sonst `$iLevel=0`). Sollte es sich um einen Fachtext mit erhöhten Anforderungen (Jura, Finanzen, etc.) und nicht um einen normalen Marketingtext (`$iExpert=0`, sonst `$iExpert=1`) handeln, ist dies gesondert zu anzugeben. Texte mit erhöhten Anforderungen können nur im 4-Augen Prinzip übersetzt werden.

Zusätzlich können allgemeine Hinweise an das Übersetzerteam und eine eigene ID zur Identifikaton / Zuordnung des Pakets mit angegeben werden.

```
$soap->createTranslationPackage($sSessionHash, $sTitle, $iSourceLanguage,
$iTargetLanguage $iLevel, $iExpert, $sHints, $sExternalID);
```

9.3 Texte zuordnen

Jeder Text, der einem Übersetzungspaket hinzugefügt wird, kann mit einem kurzen Infotext für den Übersetzer und einer externen ID (z.B. Ihrer Artikelnummer) versehen werden, um den Text später bei automatisierter Verarbeitung wieder zuordnen zu können.

```
$soap->addTextToTranslationPackage($sSessionHash,$sPackageID, $sTitle,
$sText, sHints, $sExternalID);
```

9.4 Paketstatus abfragen

Zu jedem Übersetzungspaket können Sie den Status abfragen. Neben dem Status und dem aktuellen Kosten wird kurze Zeit nach Beauftragung ggf. auch schon ein möglicher Liefertermin avisiert.

```
$soap->getTranslationPackageStatus($sSessionHash,$sPackageID);
```

9.5 Paket beauftragen

Das vorbereitete Paket wird beauftragt mit:

```
$soap->orderTranslationPackage($sSessionHash,$sPackageID);
```

9.6 Übersicht der Texte abrufen

Die Titel, IDs und externen IDs eines Paketes können abgerufen werden mit:

```
$soap-> getTranslationTextsData ($sSessionHash,$sPackageID);
```

9.7 Texte abrufen

Ein fertig übersetzter Text kann abgerufen werden mit:

```
$soap->getTranslatedText($sSessionHash,$sTextID);
```

10 Testumgebung

Zum Testen einer API-Implementierung gibt es eine Sandboxumgebung, die unter

- http://sandbox.content.de/api/...

erreichbar ist. Die Zugangsdaten entsprechen denen der Live-Umgebung. Auftraggeber Accounts ist in der Sandbox-Umgebung ein Kreditrahmen von 2.000 Euro zu Testzwecken eingeräumt. Die Sandbox-Datenbank wird in unregelmäßigen Abständen mit der Echtdatenbank überschrieben. Das Frontend der Sandbox-Umgebung ist ebenfalls nutzbar.
Eine Implementierung der API, bei der die content.de Zugriffe in einer PHP Klasse gekapselt sind, findet sich im xt:commerce-Plugin, das frei verfügbar ist unter:

- https://www.content.de/common/downloadCounter/file/xt_commerce_plugin.zip

Die hier verwendeten Klassen können als Grundlage für eine eigene Entwicklung genutzt werden. Entscheidend sind die Dateien:

- new_files/admin/includes/classes/contentde/contentdeApi.class.php
- new_files/admin/includes/classes/contentde/rpcModule.class.php

In der ersten Datei muss ggf. der Include-Path angepasst werden.
Alternativ kann auch das Wordpress Plugin von content.de als Beispiel herangezogen werden. Auch hier ist ein gekapselter API-Zugriff integriert, der in anderen Systemumgebungen problemlos (wieder-)verwendet werden kann.

- https://wordpress.org/plugins/contentde/

content.de AG
Nordstraße 14
32051 Herford

Steuernummer: 324/5723/2227
Ust-IdNr.: DE266681408
Amtsgericht Bad Oeynhausen – HRB 12246

Vorstand:
Dr. Arne-Christian Sigge
Marius Ahlers
Ralf Maciejewski

Aufsichtsratsvorsitzer:
Oliver Flaskämper

Internet: www.content.de
E-Mail: info2017@content.de